OOP - Object-Oriented Programming

OOP là một kĩ thuật lập trình cho phép tạo ra các đối tượng để trừu tượng hóa 1 đối tượng thực tế, sau đó sử dụng và tương tác với đối tượng này trong lập trình.

1. Đặc điểm của đối tượng:

Trong OOP, một đối tượng sẽ bao gồm:

- Thuộc tính (Attributes): thể hiện đặc điểm của đối tượng. Cụ thể hơn, thuộc tính là các biến lưu trữ trạng thái hoặc dữ liệu của đối tượng. Mỗi thuộc tính thường được định nghĩa trong lớp và mỗi đối tượng cụ thể sẽ có giá trị riêng cho từng thuộc tính này.
- Phương thức (Methods): thể hiện hành động của đối tượng. Cụ thể hơn, phương thức là các hàm được định nghĩa trong lớp và có thể được gọi bởi các đối tượng của lớp đó. Phương thức thường được sử dụng để thực hiện các hành động liên quan đến đối tượng, hoặc để truy cập và sửa đổi trạng thái của nó.
    - Static variables được định nghĩa bên trong một lớp nhưng bên ngoài bất kỳ phương thức nào. Chúng thường được truy cập thông qua tên lớp thay vì thông qua tên đối tượng.
    - Static Methods: Một phương thức static không cần phải tạo đối tượng mới có thể sử dụng, nó có thể được call trực tiếp qua tên lớp.

2. Tính đóng gói

Tính đóng gói (Encapsulation) là tính chất cơ bản đầu tiên thường được nhắc đến trogn OOP.

- Các dữ liệu và phương thức có liên quan được gói thành một lớp, che giấu các thông tin của lớp đó đối với bên ngoài thể hiện ở public, protected, private đối với từng thuộc tính và phương thức.
- Các dữ liệu trong lớp không thể được truy cập trực tiếp mà chỉ thông qua phương thức, giúp bảo vệ dữ liệu của lớp.
- Dễ kiểm soát quyền truy cập.

Một số từ khóa được sử dụng: private, protected, public

- public: Thuộc tính và phương thức public có thể được truy cập từ bất kỳ đâu: từ bên trong lớp, từ các lớp con, và từ bên ngoài lớp. Trong Python, các thuộc tính

và phương thức được định nghĩa mà không có dấu gạch dưới ở đầu tên được coi là public.

- protected: chỉ có thể truy cập từ các lớp con. Được định nghĩa bằng một dấu "_".
- private: chỉ có thể truy cập nội bộ trong lớp, không thể truy cập từ lớp con và các lớp bên ngoài. Được định nghĩa bằng hai dấu gạch dưới "__".

3. Tính kế thừa

TÍnh kế thừa (Inheritance): Nguyên tắc này cho phép xây dựng một lớp mới dựa trên 1 lớp đã khai báo từ trước. Lớp con có thể sử dụng lại các thuộc tính và phương thức của lớp cha mà không cần khai báo lại. Tùy thuộc vào từng ngôn ngữ cho phép việc kế thừa 1 hoặc nhiều class cha. Trong python, có thể kế thừa một con nhiều cha.

- Mục tiêu: tận dụng những thuộc tính, phương thức đã có từ lớp cha để xây dựng lớp con, tránh sự trùng lặp code.

4. Tính đa hình

Tính đa hình (Polymorphism): Tính đa hình được thể hiện bởi một phương thức, hành động có thể thực hiện theo nhiều cách khác nhau. Có 2 kiểu đa hình:

- ❖ Overload: nạp chồng, là kiểu đa hình mà trong một đối tượng có 2 phương thức cùng tên nhưng tham số truyền vào khác nhau. Vì vậy, khi call hàm đó, tùy vào tham số truyền vào, việc hàm nào được call sẽ được quyết định.
  - ➢ Giống tên nhau, cùng trong 1 đối tượng.
- ❖ Override: ghi đè, là kiểu đa hình mà khi lớp con có một phương thức cùng tên cùng tham số, nhưng thực thi khác lớp cha. Được sử dụng đi kèm tính kế thừa.
  - ➢ Số lượng tham số phải giống nhau.
  - ➢ Kiểu dữ liệu của các tham số phải giống nhau.
  - ➢ Kiểu trả về phải giống nhau hoặc phương thức trong lớp con trả về một kiểu con của lớp cha.
  - ➢ Tên phương thức giống nhau.

Tuy nhiên thực chất, Python không tồn tại overloading, có một số cách để có thể khiến cho hàm trong python hoạt động một cách gần tương tự tuy nhiên so với khái niệm overloading trong Java, nó không thực sự giống.

Ví dụ: Cùng lớp cha là Động vật, giả sử có 2 lớp con là Chó và Mèo. Động vật có phương thức kêu()

- Trong lớp cho và mèo đều có kêu(), nhưng chó kêu kiểu khác và mèo kêu kiểu khác, đó là override.

- Trong lớp chó, ngoài kêu() còn có 1 phương thức khác cùng tên nhưng có thêm tham số ăn no = true thì kêu to hơn. Đó là overload.

5. Tính trừu tượng

TÍnh trừu tượng (Abstraction): tổng quát hóa phương thức của đối tượng không quan tâm phương thức thực hiện như thế nào, chỉ quan tâm có những phương thức nào, có thể làm được gì (được thể hiện bằng interface). Tức là nó chỉ có tên phương thức và không có phần body bên trong, khi class nào implement interface, phương thức sẽ được thực hiện (do ghi đè).

- Phương thức trừu tượng là một phương thức chỉ có khai báo và không có định nghĩa.
- Một lớp được gọi là lớp trừu tượng chỉ khi nó có ít nhất một phương thức trừu tượng.
- Khi kế thừa một lớp trừu tượng làm lớp cha cho lớp con, lớp con phải định nghĩa tất cả các phương thức trừu tượng có trong lớp cha. Nếu không làm như vậy thì lớp con cũng sẽ trở thành lớp trừu tượng tự động.
- Python mặc định không hỗ trợ lớp trừu tượng và phương thức trừu tượng, vì vậy có một gói gọi là ABC (lớp cơ sở trừu tượng) mà bằng cách đó chúng ta có thể làm cho một lớp hoặc phương thức trở thành trừu tượng.